Multi-Scale Patch-Based Representation Learning for Image Anomaly Detection and Segmentation

Chin-Chia Tsai[1], Tsung-Hsuan Wu[1], and Shang-Hong Lai[1,2]

s108065530@m108.nthu.edu.tw, th.wu@mx.nthu.edu.tw, shlai@microsoft.com

[1]Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan
[2]Microsoft AI R&D Center, Taipei, Taiwan

Abstract

Unsupervised representation learning has been proven to be effective for the challenging anomaly detection and segmentation tasks. In this paper, we propose a multi-scale patch-based representation learning method to extract critical and representative information from normal images. By taking the relative feature similarity between patches of different local distances into account, we can achieve better representation learning. Moreover, we propose a refined way to improve the self-supervised learning strategy, thus allowing our model to learn better geometric relationship between neighboring patches. Through sliding patches of different scales all over an image, our model extracts representative features from each patch and compares them with those in the training set of normal images to detect the anomalous regions. Our experimental results on MVTec AD dataset and BTAD dataset demonstrate the proposed method achieves the state-of-the-art accuracy for both anomaly detection and segmentation.

1. Introduction

Anomaly detection has been a critical and common problem for the manufacturing industries. Due to the limitation of human attention span, consistently maintaining high quality for the manufactured products through human visual inspection is almost impossible and infeasible. Thus, automatic anomaly detection is highly demanded for intelligent manufacturing.

In computer vision, anomaly detection aims to decide whether an image is a normal or an abnormal sample, usually providing an anomaly score as a reference for the anomaly decision. Since anomalous data is either inaccessible or insufficient and anomalies may contain unexpected patterns, the anomaly detection problem is usually formulated as a one-class learning setting [4], [19], [27], i.e., only normal data is available for training.

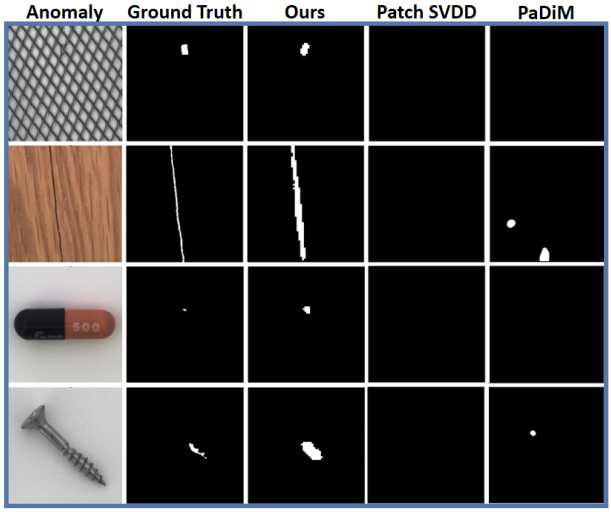

Figure 1. Examples of challenging defects in MVTec AD dataset. From top row to bottom row are Grid, Wood, Capsule and Screw respectively. Our predicted masks compared with those by Patch SVDD [27] and PaDiM [7] show that our model outperforms those methods on detecting these challenging defects.

To further strengthen the confidence of trusting the results of anomaly detection, localizing the anomalous regions, which are called defects, in the images at pixel level is helpful to provide more precise and interpretable results. This task is known as anomaly localization, or anomaly segmentation. However, to achieve high-precision anomaly detection and localization without using labeled training data is still a challenging task.

Anomaly detection and segmentation aim to distinguish between normal images and anomalous images on image-

level and pixel-level, respectively. Since the anomalous regions are usually tiny in the whole image, anomaly segmentation, often solved by splitting images into small patches, is very challenging especially when the anomalous regions are very small or the anomalous appearance is not very clear. If the model is trained with focus on small anomalies [6], [8], [25], small anomalous regions may be more accurately segmented but it could possibly increase the chance of mis-classifying normal regions to anomalous regions at the same time, thus making FPR (False Positive Rate) increased. Therefore, we need to have a very powerful feature representation learned from normal images only for both the anomaly detection and anomaly segmentation tasks .

A good feature representation learning method can not only provide excellent performance on anomaly segmentation but also make the model robust against unseen anomalies. In this work, we aim to develop a novel feature representative learning framework for anomaly detection and segmentation to extract representative features from multi-scale patches, thereby obtaining the global and local context of an image at the same time for better representation learning.

Our main contributions are listed as follows:

1. We propose a multi-scale patch-based architecture for different levels of representation learning. We show that considering the global and local context of an image at the same time leads to better representation learning. Furthermore, our model is scalable for different sizes of patches, which makes it adaptable to various application scenarios.

2. We introduce K-means clustering and cosine similarity to develop a new loss function for better feature representation learning from normal samples, which is evident from visualization of concentrated distribution of the learned features computed from normal images.

3. The improved feature representation learning method leads to superior performance for image anomaly detection and segmentation on MVTec AD dataset and BTAD dataset compared to the state-of-the-art methods.

2. Related Work

2.1. Reconstruction-based methods

Previous deep learning methods for anomaly detection and segmentation are usually based on reconstruction-based neural network architectures, such as autoencoders (AE) [3], [5], [11], [15], [17], variational autoencoders (VAE) [14], [23], and generative adversarial networks (GAN) [1], [20]. These architectures are trained to reconstruct normal training images accurately. If an anomalous image comes as an input, then it is supposed to output a bad reconstructed image. The anomaly score is calculated from the reconstruction error between the input image and its reconstruction. This idea is intuitive and interpretable. However, AE can sometimes yield good reconstruction for anomalous images unexpectedly due to the generalization ability of machine learning models. [11] proposed a memory-augmented autoencoder (MemAE) to suppress the generalization ability of AE by encouraging the memory contents to represent the prototypical elements of the normal data. Yet MemAE is unfavorable to anomaly segmentation owing to the low resolution of its reconstructed images. [17] improved this drawback by introducing feature compactness loss and feature separateness loss. But still, the resolution of reconstructed images is not enough to achieve anomaly segmentation for high-resolution images, which is vital for many industrial applications.

2.2. Feature-based methods

Recently, more and more feature-based anomaly detection methods have been proposed. The main idea of feature-based methods is to extract meaningful feature vectors for describing the entire image [2], [19] or the patches of the image [4], [6], [24], [27]. The anomaly score will then be calculated by the distance between the representing feature vectors of the normal training data and the embedding vectors of a testing image. Because the appearances of normal and anomalous images are usually very different, the distance between the corresponding features is supposed to be large especially when the testing image is anomalous.

Deep SVDD [19] used a deep neural network in place of the kernel function in SVDD [22], a classical one-class classification method. [27] extended deep SVDD to a patch-level anomaly detection, and proposed the Patch SVDD method, which is also the baseline method of our work. Patch SVDD enables SVDD to do anomaly segmentation and at the same time improves the anomaly detection performance significantly. It uses SVDD loss to train an encoder to gather semantically similar patches and makes the embeddings of adjacent patches still distinguishable enough by adopting the self-supervised learning method proposed by [10], which trains an encoder and classifier pair to predict the relative position of two patches. Nevertheless, the idea of mapping the features of adjacent patches together benefits the cases of patches with similar structures. Moreover, predicting relative position of two patches can be confusing by cases with texture images. To overcome the above two problems, [27] proposed to increase the weight of SVDD loss for texture images and decrease it for object images. Unfortunately, information extracted from the images tends to be insufficient by only adjusting the weights of losses.

3. Proposed Method

Inspired by Patch SVDD in [27], this work aims to learn more representative features from normal images. Based on a multi-scale patch architecture, our model is trained to extract representative patch features from normal images.

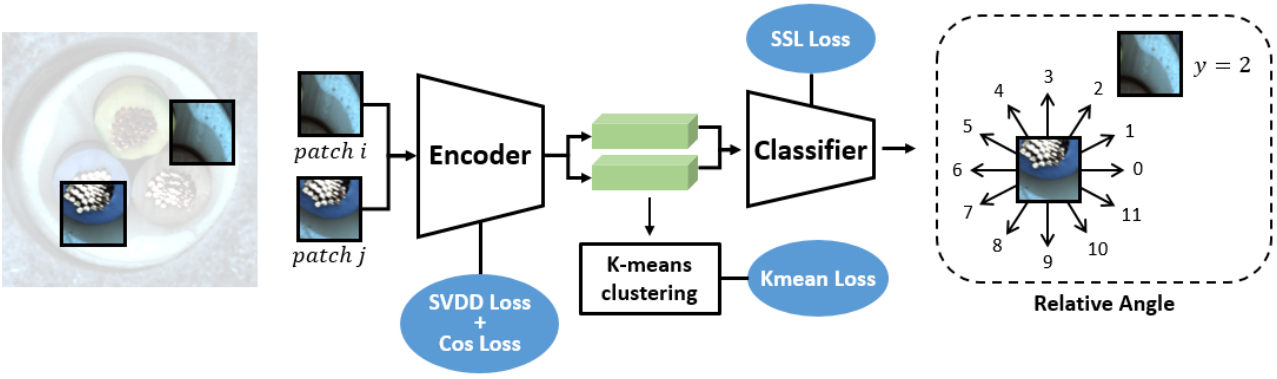

Figure 2. Our framework uses multi-scale patches for feature extraction from encoders Enc_{64}, Enc_{32}, and Enc_{16}, which are pretrained on ImageNet with VGG16 architecture. The features extracted from each of these encoders will be clustered by using K-means clustering method. Meanwhile, they will be sent into a classifier C for predicting relative positions between patches. Note that the patch sizes are 64, 32, and 16, respectively, and all the three encoders follow the same training flow.

Figure 2 depicts an overview of the proposed framework. The whole framework is mainly composed of 3 encoders with different architectures and 3 classifiers. The overall flow of the proposed anomaly detection and segmentation system will be described in details subsequently.

3.1. Training Stage

In our training process, we first select two patches of size 64×64 and use the same selection method for patches of size 32×32 and size 16×16 from the same image several times. The 3 different sizes of patches will then go through the same work flow except that the encoders are with different architectures. Here the 3 encoders are denoted as Enc_{64}, Enc_{32}, and Enc_{16} respectively.

Take the patches of size 64×64 for example. The selected patches will be sent into Enc_{64}, which is pretrained on ImageNet [9] with VGG16 [21] architecture. The feature embeddings of these patches encoded by Enc_{64} will then be clustered by applying K-means clustering [12] method. Meanwhile, the features will be sent into a classifier C to predict the relative angles between the patches.

3.2. Objective Functions

Our network is trained with four different objective functions, which are SVDD loss, Cos loss, SSL loss, and Kmean loss. The details of these objective functions will be described in the following subsections. Here Enc_θ is denoted as the encoders, where $\theta \in \{64, 32, 16\}$ denotes the patch size. The approach of selecting the patches in the four losses will be explained in details in 3.3.

3.2.1 SVDD Loss and Cos Loss

We follow the concept of gathering semantically similar patches by SVDD loss in [27]. By sampling spatially adjacent patches, the encoder is trained to minimize the L2-distance between their features. Besides the SVDD loss, we expect the two patches with larger distance to be semantically less similar. Thus, we further add the Cos loss to strengthen the information extracted from the patches. Equation 1 and 2 show the SVDD loss and the Cos loss, respectively.

$$\mathcal{L}_{\text{SVDD}} = \sum_{(i,j)\in N} \|Enc_\theta(P_i) - Enc_\theta(P_j)\|_2, \tag{1}$$

and

$$\begin{aligned}\mathcal{L}_{\text{Cos}} = \sum_{(i,j,k)\in T} & Sim\_cos(Enc_\theta(P_i), Enc_\theta(P_k)) \\ & - Sim\_cos(Enc_\theta(P_i), Enc_\theta(P_j)),\end{aligned} \tag{2}$$

where N denotes the set of pairs of neighboring patches, T denotes a set of triplets of patches (P_i, P_j, P_k)'s with P_j and P_i closer than P_k and P_i, and $Sim\_cos(\mathbf{u}, \mathbf{v})$ is the cosine similarity between feature vectors $\mathbf{u}$ and $\mathbf{v}$.

3.2.2 SSL Loss

We also follow [27] to utilize the self-supervised learning method in [10]. Yet we extend the concept of predicting the relative positions into relative angles, for the purpose of helping the model predict more accurate direction between two neighboring patches. The classifier C is trained to predict the relative angles between the selected two neighbor-

ing patches P_i and P_j.

$$\mathcal{L}_{\text{SSL}} = \sum_{(i,j)\in N} CrossEntropy(y, C(Enc_\theta(P_i), Enc_\theta(P_j))), \tag{3}$$

where $y \in \{0, 1, 2, ..., 11\}$ is the index of ground-truth angles of P_j relative to P_i and the set corresponds to the angles $\{0°, 30°, 60°, ..., 330°\}$.

3.2.3 Kmean Loss

In order to learn better feature representation of normal patterns, we adopt K-means clustering [12] to gather patches with similar patterns. Since the embeddings of the patches in the same cluster are expected to be closer to the center of that cluster as much as possible, we define the K-means loss as follows:

$$\mathcal{L}_{\text{Kmeans}} = \sum_{r} \min_{k} \|Enc_\theta(P_r) - c_k\|_2, \tag{4}$$

where c_k are the centers of the clusters. Here we update c_k for every 5 epochs.

3.2.4 Overall Loss

Finally, the overall loss for optimizing our model is given by Equation 5.

$$\mathcal{L}_{\text{all}} = \lambda(L_{\text{SVDD}} + L_{\text{Cos}}) + L_{\text{SSL}} + L_{\text{Kmeans}}, \tag{5}$$

where λ is the hyper-parameter in the objective function for the model training. If the data has similar pattern for all patches, then the value λ can be set to a larger value for relying much more on the semantically similarity among patches, and vice versa. However, here λ can be set to the same value regardless of the image structure in our model due to the sufficient information extracted from the embedding features.

3.3. Patch Selection

The strategy of the patch selection in our model training is different for different loss functions. Figure 3 depicts an example of selecting patches for different loss functions. For the SVDD loss and Cos loss, we first randomly select a patch P_i from the image. Then a patch P_j is randomly selected from the ± 4 pixels adjacent patches to P_i if the patch size is 64, ± 2 pixels adjacent patches to P_i if the patch size is 32, and ± 1 pixel adjacent patches to P_i if the patch size is 16. Patch P_k is randomly selected from the patches which are further away from P_i. On the other hand, the process of the patch selection for SSL loss is to randomly select a patch P_m first, and then select a patch P_n nearly a patch size away from P_m along a direction randomly selected from the 12 pre-defined angles. Finally, Kmean loss simply selects patches P_r from the image randomly.

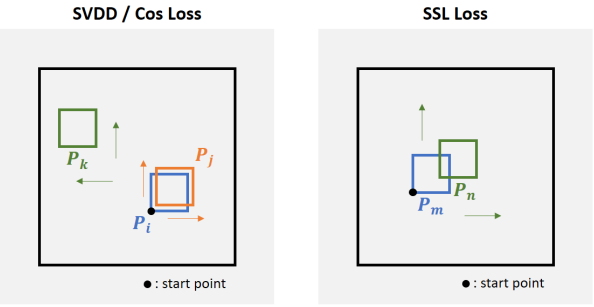

Figure 3. An example of patch selection process. The left side of the figure shows how patches are selected for SVDD loss and Cos loss and the right side demonstrates SSL loss.

For the whole training process, the total number of patches, patch pairs, or patch triplets selected for each loss is set to 100 for each image in our implementation.

3.4. Inference Stage

Figure 4 depicts the process of our inference phase. A test image will first be split into overlapped patches with patch size $\theta \in \{64, 32, 16\}$ and with stride d_θ. In our experiments, $d_{64} = 16$ and $d_{32} = d_{16} = 4$. Then the trained encoders Enc_{64}, Enc_{32}, and Enc_{16} extract features from these patches. For each patch P_r^θ, we evaluate its abnormality by calculating the shortest L2-distance between its feature embedding and all the normal feature embeddings in the training dataset by

$$\mathcal{D}_\theta(P_r^\theta) = \min_{t\in T^\theta} \left\|Enc_\theta(P_r^\theta) - Enc_\theta(P_t)\right\|_2, \tag{6}$$

where T^θ is the index set of all patches of the training images with patch size θ. Next, we picture the anomaly map for each θ by making the above patch-wise calculated anomaly scores distributed to the pixels by averaging the scores overlapped on the same pixels, i.e., the value of the anomaly map AM_θ at the (i, j)-pixel is defined by

$$AM_\theta(i, j) = \underset{a\in I_{i,j}^\theta}{mean}(D_\theta(P_a^\theta)), \tag{7}$$

where $I_{i,j}^\theta$ is the index set of the patches covering the (i, j)-pixel. After that, we aggregate the three maps corresponding to different patch sizes by using element-wise addition to obtain the final anomaly map AM, i.e.,

$$AM(i, j) = \sum_{\theta} AM_\theta(i, j), \tag{8}$$

is used as our final anomaly segmentation result.

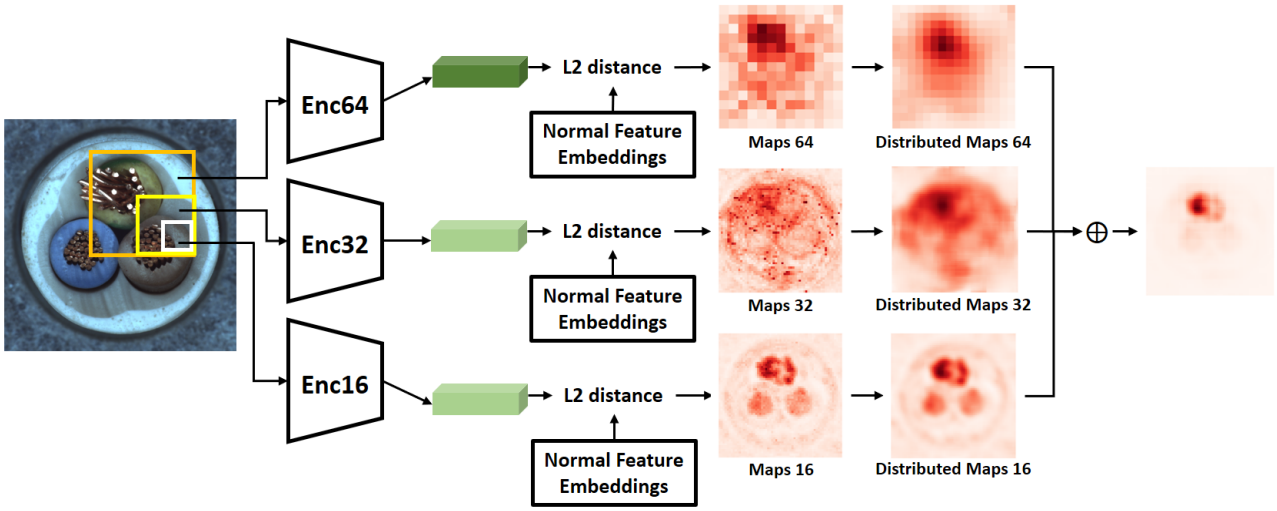

Figure 4. The detailed process of our inference phase. Features of different sizes of patches are extracted from Enc_{64}, Enc_{32}, and Enc_{16} respectively. Note that the anomaly maps here are all 256×256.

For the anomaly detection, we use a sliding window over the anomaly map with window size 8×8 and stride 4. Then we evaluate a local score for each window by averaging the scores inside it. The final anomaly score of the image is then defined by the maximum of all the local scores as follows:

$$Score = \max_{(s_1,s_2)\in S} \frac{1}{64} \sum_{i=0}^{7} \sum_{j=0}^{7} AM(i+s_1, j+s_2), \tag{9}$$

where S is the set of the top-left positions of the windows. We adopt this strategy for anomaly detection due to its ability to alleviate the influence of outlier pixels with high anomaly scores for normal patches, which effectively reduces the false positives.

4. Experiments

4.1. Datasets and Evaluation Metrics

We conduct our experiments mainly on the MVTec AD dataset [3], which is a dataset for benchmarking anomaly detection methods on industrial inspection. The dataset contains over 5,000 high-resolution ($700 \times 700 \sim 1024 \times 1024$ $pixels$) images with 5 texture classes and 10 objects classes. Each class is composed of defect-free training images and testing images with various types of defects as well as defect-free images. The testing data also provides pixel-precise annotations of all anomalies.

We also perform experiment on BTAD (beanTech Anomaly Detection dataset) dataset, which is recently released by [16]. It contains 2,830 images with 3 different classes. The resolution of these 3 classes are 1600×1600, 600×600, and 800×600, respectively. Each class is composed of defect-free training images and testing images, like MVTech AD dataset, except that the defect types are not illustrated.

Similar to most recent works and the state-of-the-art methods, we adopt Area Under the Receiver Operating Characteristics (AUROC) as the evaluation metric for fair comparison. The threshold is determined by the point corresponding to the maximal F1-score of the precision-recall-curve between the anomaly scores and the ground truth. We also compute another evaluation metric suggested by [3], per-region-overlap score(PRO-score), which weights ground-truth anomalous regions equally regardless of the sizes of the regions. The main metric AUROC is biased in favor of large anomalous region, whereas for PRO-score, a large correctly segmented region cannot make up for wrongly segmented minor ones. Same as [3] and other papers, we produce a binary mask which indicates whether a pixel is an anomaly by giving a threshold to anomaly scores. The PRO-score is calculated as the average proportion of the pixels detected as anomaly in each ground truth anomalous regions. Then the measure of the PRO-score will be the normalized value of the integral across per-pixel false-positive rates from 0 to 0.3. A high PRO-score indicates that both large and minor anomalous regions are well-localized.

4.2. Experimental Comparison

Our model is implemented in PyTorch. We resize images in MVTec AD dataset to 256×256 and images in BTAD to 512×512. Here λ is empirically set to 10^{-3} and batch size is set to 64 for all classes in both datasets.

Table 1. Comparison of our models with the SOTA methods for the image-level anomaly detection performance on MVTec AD dataset. The results are reported with AUROC%.

Class	CutPaste [13]	STPM [24]	InTra [18]	SPADE [6]	Patch SVDD [22]	PaDiM [7]	**Ours**
Carpet	93.1	-	**98.8**	-	92.9	-	93.4
Grid	99.9	-	**100.0**	-	94.6	-	**100.0**
Leather	**100.0**	-	**100.0**	-	90.9	-	99.3
Tile	93.4	-	**98.2**	-	97.8	-	96.2
Wood	98.6	-	98.0	-	96.5	-	**99.7**
All Texture Classes	97.0	-	**99.0**	-	94.5	-	97.7
Bottle	98.3	-	**100.0**	-	98.6	-	**100.0**
Cable	80.6	-	84.2	-	90.3	-	**98.8**
Capsule	96.2	-	86.5	-	76.7	-	**97.2**
Hazelnut	97.3	-	95.7	-	92.0	-	**99.6**
Metal_nut	**99.3**	-	96.9	-	94.0	-	97.8
Pill	92.4	-	90.2	-	86.1	-	**97.7**
Screw	86.3	-	**95.7**	-	81.3	-	94.1
Toothbrush	98.3	-	99.7	-	**100.0**	-	**100.0**
Transistor	95.5	-	95.8	-	91.5	-	**98.9**
Zipper	99.4	-	99.4	-	97.9	-	**99.5**
All Object Classes	94.3	-	94.4	-	90.8	-	**98.4**
All classes	95.2	95.5	95.9	85.5	92.1	97.9	**98.1**

The optimizer is Adam with the learning rate 10^{-5}. Note that Patch SVDD [27] used different λ values for different classes whereas we use the same λ value for all experiments in the paper.

We compare the performance of our model with several state-of-the-art anomaly detection methods on MVTec AD dataset and BTAD dataset. Table 1 and Table 2 show the superior overall accuracy of our methods compared with the SOTA methods, including CutPaste [13], STPM [24], InTra [18], SPADE [6], Patch SVDD [22], and PaDiM [7], on anomaly detection and anomaly segmentation, respectively.

Table 2. Comparison of our models with the SOTA methods for the pixel-level anomaly localization performance on MVTec AD dataset. The results are reported with AUROC%.

Class	CutPaste [13]	STPM [24]	InTra [18]	SPADE [6]	Patch SVDD [27]	PaDiM [7]	**Ours**
Carpet	98.3	98.8	**99.2**	97.5	92.6	99.1	98.4
Grid	97.5	99.0	**99.4**	93.7	96.2	97.3	98.5
Leather	**99.5**	99.3	**99.5**	97.6	97.4	99.2	99.1
Tile	90.5	**97.4**	94.4	87.4	91.4	94.1	94.4
Wood	95.5	97.2	90.5	88.5	90.8	94.9	**97.5**
All Texture Classes	96.3	**98.3**	96.6	92.9	93.7	96.9	97.6
Bottle	97.6	**98.8**	97.1	98.4	98.1	98.3	98.6
Cable	90.0	95.5	93.2	97.2	96.8	96.7	**98.2**
Capsule	97.4	98.3	97.7	**99.0**	95.8	98.5	97.9
Hazelnut	97.3	98.5	98.3	**99.1**	97.5	98.2	97.8
Metal_nut	93.1	97.6	93.3	98.1	98.0	97.2	**99.1**
Pill	95.7	97.8	98.3	96.5	95.1	95.7	**98.8**
Screw	96.7	98.3	**99.5**	98.9	95.7	98.5	98.5
Toothbrush	98.1	98.9	**99.0**	97.9	98.1	98.8	**99.0**
Transistor	93.0	82.5	96.1	94.1	97.0	97.5	**97.7**
Zipper	**99.3**	98.5	99.2	96.5	95.1	98.5	98.6
All Object Classes	95.8	96.5	97.2	97.6	96.7	97.8	**98.4**
All classes	96.0	97.0	97.0	96.5	95.7	97.5	**98.1**

Note that our method achieves state-of-the-art results on several classes and provides the highest average AUROC among all classes for both anomaly detection and segmentation tasks. It is obvious that the previous SOTA methods usually perform not well on one or two classes. However, our method performs consistently well for all classes in this experiment.

Table 3 shows the comparison of our models with the SOTA methods which also conduct the PRO-score calculation. Our model performs better on most of the classes, especially for the class $Transistor$. The results prove that our proposed method segment both large and minor anomalous regions well.

Table 3. Comparison of our models with the SOTA methods for the pixel-level anomaly localization performance on MVTec AD dataset. The results are reported with PRO-score%.

Class	STPM [24]	U-Student [4]	DFR [26]	SPADE [6]	PaDiM [7]	**Ours**
Carpet	95.8	87.9	93.0	94.7	**96.2**	92.7
Grid	96.6	95.2	93.0	86.7	94.6	**97.9**
Leather	98.0	94.5	97.0	97.2	97.8	**99.2**
Tile	92.1	**94.6**	79.0	75.9	86.0	88.8
Wood	93.6	91.1	91.0	87.4	91.1	**96.2**
All Texture Classes	**95.2**	92.7	90.6	88.4	93.2	95.0
Bottle	95.1	93.1	93.0	**95.5**	94.8	95.3
Cable	87.7	81.8	81.0	90.9	88.8	**96.7**
Capsule	92.2	96.8	97.0	93.7	93.5	**97.8**
Hazelnut	94.3	96.5	97.0	95.4	92.6	**97.8**
Metal_nut	**94.5**	94.2	90.0	94.4	85.6	88.8
Pill	**96.5**	96.1	96.0	94.6	92.7	96.1
Screw	93.0	94.2	96.0	96.0	94.4	**98.3**
Toothbrush	92.2	93.3	93.0	93.5	93.1	**94.4**
Transistor	69.5	66.6	79.0	87.4	84.5	**95.0**
Zipper	95.2	95.1	90.0	92.6	95.9	**97.0**
All Object Classes	91.0	90.8	91.0	93.4	91.6	**95.7**
All classes	92.1	91.4	91.0	91.7	92.1	**95.5**

We also compare our model with VT-ADL [16] and Patch SVDD [27] on the recently released BTAD dataset. Note that [16] only reported its anomaly segmentation accuracy with PRO-score and PR-AUC. For comparison with our method, we use the code of the works [16] and [27] released in Github and output both the anomaly detection and segmentation AUROC results on BTAD dataset. Table 4 and Table 5 show that [16] only performs well on Product01 for anomaly detection task. On the contrary, our method achieves 95.1% average AUROC on anomaly detection and 97.7% average AUROC on anomaly segmentation for all the three product types. The results show the high accuracy and generalization of applying our method to detect different styles and resolutions of defects.

4.3. Ablation Study

We conduct various ablation studies on the MVTec AD dataset to provide deeper insight of the proposed method. Multi-scale architecture is the main characteristics of our model. From Table 6, we prove that with different scales of patches, representative features are extracted from the

Table 4. Comparison of results by applying VT-ADL[16], Patch SVDD [27] and our models for the image-level anomaly detection on BTAD dataset. The results are reported with AUROC%.

Product	VT-ADL[16]	Patch SVDD [27]	Ours
Product01	97.6	96.3	**100.0**
Product02	71.0	70.3	**85.3**
Product03	82.6	91.1	**100.0**
All Products	83.7	85.9	**95.1**

Table 5. Comparison of results by applying VT-ADL[16], Patch SVDD [27] and our models for the pixel-level anomaly segmentation on BTAD dataset. The results are reported as AUROC%.

Product	VT-ADL[16]	Patch SVDD [27]	Ours
Product01	76.3	94.9	**97.3**
Product02	88.9	92.7	**96.8**
Product03	80.3	91.7	**99.0**
All Products	81.8	93.1	**97.7**

Table 6. Study of the image-level anomaly detection and the pixel-level anomaly segmentation performance with different scales on MVTec AD dataset. Here single-scale denotes patch of size 64, 2-scale denotes size 64 and 32, and 3-scale denotes size 64, 32, and 16. The results are reported with AUROC%.

Class	All Texture Classes		All Object Classes		All Classes	
Task	**det.**	**seg.**	**det.**	**seg.**	**det.**	**seg.**
Ours (single-scale)	81.5	84.6	93.2	95.2	89.3	91.7
Ours (2-scale)	97.0	95.8	97.1	97.9	97.1	97.2
Ours (3-scale)	**97.7**	**97.6**	**98.4**	**98.4**	**98.1**	**98.1**

images. The improvement on the experimental evaluation on different texture classes is especially significant from single-scale to multi-scale, especially on subtle defect regions.

Table 7 shows our ablation study for each loss function. It is obvious that without SSL loss, the performance drops significantly for all classes, which justifies the importance of the self-supervised learning component. Besides, we notice that removing the Cos loss or K-means loss cause considerable degradation on the accuracy, especially for texture classes.

4.4. t-SNE Visualization

t-SNE is commonly used as an dimensionality reduction tool for the visualization of high-dimensional feature distributions. It is also frequently used in anomaly detection tasks to visualize the distributions of learned features. Here we show the t-SNE plots of the learned features from our model in the top row of Figure 5. It can obvious that the

Table 7. Study of the image-level anomaly detection and the pixel-level anomaly segmentation performance with different losses on MVTec AD dataset. The results are reported with AUROC%.

Class	All Texture Classes		All Object Classes		All Classes	
Task	**det.**	**seg.**	**det.**	**seg.**	**det.**	**seg.**
Ours (w/o SVDD)	96.0	95.5	96.0	97.3	96.0	96.7
Ours (w/o Cos)	94.8	93.4	97.8	98.0	96.8	96.5
Ours (w/o SSL)	89.9	93.8	88.9	95.7	89.3	95.0
Ours (w/o Kmean)	94.8	93.9	97.2	98.2	96.4	96.8
Ours	**97.7**	**97.6**	**98.4**	**98.4**	**98.1**	**98.1**

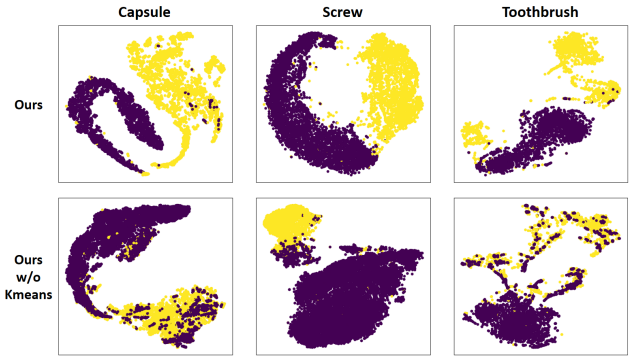

Figure 5. t-SNE plots of the learned features of our full model and our model without Kmeans loss. Purple points and yellow points denote normal patches and anomalous patches, respectively. The plots show that our model can better separate anomaly from normal samples.

normal patches and the anomalous patches are well separated in the learned feature space. This suggests that our model learns the feature representation very well from normal images only. Note that each point here is a patch from a testing image. To emphasize the importance of K-means clustering method, we also conduct the t-SNE visualization of the learned features without Kmeans loss in the bottom row of Figure 5. The comparisons between the plots demonstrate the contribution of K-means clustering method to our model.

4.5. Qualitative Results

In this section, we show some examples of anomaly segmentation results for different classes in Figure 6 and Figure 7 to demonstrate the performance of our model. We calculate F1-score as the threshold of every anomaly map like most of the previous works. Each anomaly score of the pixel larger than the threshold is regarded as anomalous pixel and set to 1. Then the mask composed of 0 and 1 generates our predicted mask. It can be shown from the images that regardless of the sizes, shapes, or positions of the defects, all predicted defects produced by our

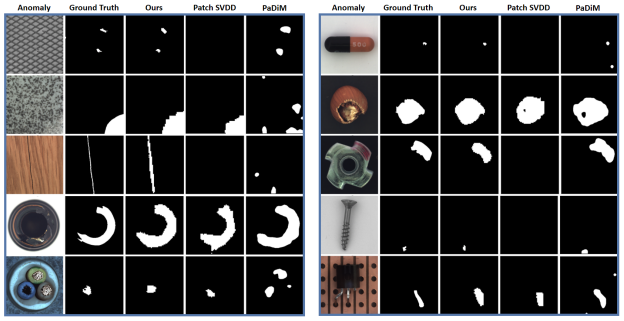

Figure 6. Examples of different classes in MVTec AD dataset. From top row to bottom row are Grid, Tile, Wood, Bottle, and Cable on the left side, and Capsule, Hazelnut, Metal_nut, Screw, and Transistor on the right side respectively. We compare our predicted masks with Patch SVDD [27] and PaDiM [7].

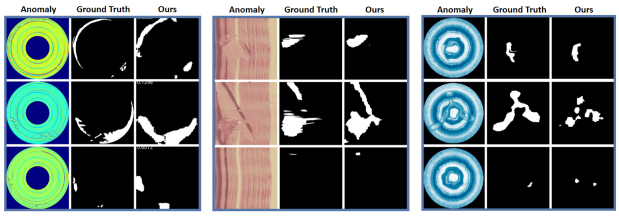

Figure 7. Examples of the 3 products of BTAD dataset. Product01 to Product03 are shown from the left side to the right side. Here we show the predicted masks obtained from our proposed method.

proposed method are properly localized. Although some shapes of the masks seem not very accurate, the predicted masks still justify that our model performs well on detecting tiny defects. Compared with our predicted masks in Figure 6, [27] misses defects of small anomalous regions and texture classes, and [7] has some false positive detection on a few classes.

4.6. Analysis of Failure Cases

In this section, we give deeper insight in some failure cases in the MVTec AD dataset and the BTAD dataset. We concentrate on texture classes here since our model performs worse for some texture classes.

4.6.1 MVTec AD Dataset

From Figure 8, we can observe that our model fails to detect some scratch defect type with thin and tiny or thin and long scratches for the classes Carpet. Also, our model struggles to detect the defect type $thread\_side$ of the class Screw. And for the class Tile, our model failed to detect some slightly rough cases for the defect type $rough$, as shown in the figure.

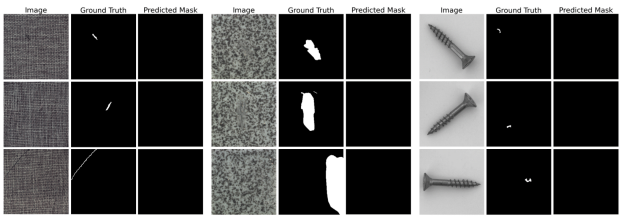

Figure 8. Examples of failure cases in defective testing images in the MVTec AD dataset. From left to right are Carpet, Tile, and Screw classes, respectively.

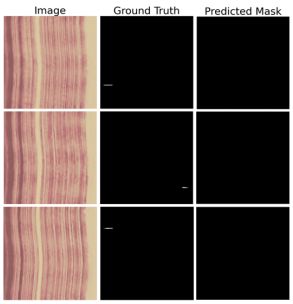

Figure 9. Examples of failed cases in defective testing images of Product02 in the BTAD dataset.

4.6.2 BTAD Dataset

For the experimental results on BTAD dataset, our model performs slightly worse for Product02 in the image-level anomaly detection task. We dig into the failed images and find out that there are 96 out of 200 defective test images containing only tiny or thin scratches, as shown in Figure 9. These challenging scratches are undetected by our model even with 3-scale patches and patch size 16. These tiny or thin anomalous regions are even smaller and difficult to detect from the image normalized to size 256x256, which is used as input to our model.

5. Conclusion

We proposed a multi-scale patch-based framework of image representation learning for anomaly detection. Our experimental results prove that considering the global and local context of an image at the same time leads to excellent representation learning for image anomaly detection. Moreover, our multi-scale system is capable of detecting anomalous regions of different sizes. Our experimental results demonstrate that the proposed method achieves SOTA accuracy on the benchmark datasets for image anomaly detection and segmentation.

References

[1] Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Asian Conference on Computer Vision*, pages 622–637. Springer, 2018.

[2] Liron Bergman and Yedid Hoshen. Classification-based anomaly detection for general data. In *International Conference on Learning Representations (ICLR)*, 2020.

[3] Paul Bergmann, Michael Fauser, David Sattlegger, and C. Steger. Mvtec ad — a comprehensive real-world dataset for unsupervised anomaly detection. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9584–9592, 2019.

[4] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4182–4191, 2020.

[5] Paul Bergmann, Sindy Löwe, Michael Fauser, David Sattlegger, and Carsten Steger. Improving unsupervised defect segmentation by applying structural similarity to autoencoders. *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2019.

[6] Niv Cohen and Yedid Hoshen. Sub-image anomaly detection with deep pyramid correspondences. *arXiv:2005.02357*, 2020.

[7] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. Padim: a patch distribution modeling framework for anomaly detection and localization. *ICPR*, 2020.

[8] David Dehaene and Pierre Eline. Anomaly localization by modeling perceptual features. *arXiv:2008.05369*, 2020.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[10] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1422–1430, 2015.

[11] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton Van Den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2019.

[12] Xin Jin and Jiawei Han. *K-Means Clustering*, pages 563–564. Springer US, Boston, MA, 2010.

[13] Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[14] Wenqian Liu, Runze Li, Meng Zheng, Srikrishna Karanam, Ziyan Wu, Bir Bhanu, Richard J. Radke, and Octavia Camps. Towards visually explaining variational autoencoders. *arXiv:1911.07389*, 2020.

[15] Shuang Mei, Hua Yang, and Zhouping Yin. An unsupervised-learning-based approach for automated defect inspection on textured surfaces. *IEEE Transactions on Instrumentation and Measurement*, 67(6):1266–1277, 2018.

[16] Pankaj Mishra, Riccardo Verk, Daniele Fornasier, Claudio Piciarelli, and Gian Luca Foresti. VT-ADL: A vision transformer network for image anomaly detection and localization. In *30th IEEE/IES International Symposium on Industrial Electronics (ISIE)*, June 2021.

[17] Hyunjong Park, Jongyoun Noh, and Bumsub Ham. Learning memory-guided normality for anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14372–14381, 2020.

[18] Jonathan Pirnay and Keng Chai. Inpainting transformer for anomaly detection. *arXiv:2104.13897*, 2021.

[19] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4393–4402. PMLR, 10–15 Jul 2018.

[20] Mohammad Sabokrou, Mohammad Khalooei, Mahmood Fathy, and Ehsan Adeli. Adversarially learned one-class classifier for novelty detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3379–3388, 2018.

[21] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[22] David M.J. Tax and Robert P.W. Duin. Support vector data description. *Machine Learning*, 54:45–66, 2004.

[23] Shashanka Venkataramanan, Kuan-Chuan Peng, Rajat Vikram Singh, and Abhijit Mahalanobis. Attention guided anomaly localization in images. *arXiv:1911.08616*, 2020.

[24] Guodong Wang, Shumin Han, Errui Ding, and Di Huang. Student-teacher feature pyramid matching for unsupervised anomaly detection. *arXiv:2103.04257*, 2021.

[25] Lu Wang, Dongkai Zhang, Jiahao Guo, and Yuexing Han. Image anomaly detection using normal data only by latent space resampling. *Applied Sciences*, 10(23), 2020.

[26] Jie Yang, Yong Shi, and Zhiquan Qi. Dfr: Deep feature reconstruction for unsupervised anomaly segmentation. *ArXiv*, abs/2012.07122, 2020.

[27] Jihun Yi and Sungroh Yoon. Patch svdd: Patch-level svdd for anomaly detection and segmentation. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, November 2020.